

**CLAIMS**

What is claimed is:

1           1.       A method for optimizing the use of a plurality of processors when compiling a  
2 program in a computer system, the method comprising the steps of:

- 3           (a)     providing a list of directories and a list of processors;
- 4           (b)     assigning a directory to a next available processor in an ordered manner to  
5 allow the next available processor to compile at least one file within the directory; and  
6           (c)     repeating step (b) to ensure that the maximum number of directories can be  
7 compiled.

1           2.       The method of claim 1 wherein the assigning step (b) further includes the step  
2 of (b1) obtaining a directory in which all dependencies have been satisfied.

1           3.       The method of claim 1 wherein the assigning step (b) further includes the step  
2 of (b1), updating the list of processors and the list of directories based upon the assignment of  
3 the directory.

1           4.       The method of claim 1 wherein the assigning step (b) further includes the step  
2 of (b1) providing a directory update mechanism for assigning the directories in the ordered  
3 manner.

1           5.       The method of claim 4 wherein the providing an update mechanism step (b1)

2 further comprises the steps of:

3 (b11) providing an array of dependency changes; and

4 (b12) merging the dependency changes array with a master array of changes.

1 6. The method of claim 5 wherein the merging step (b12) comprises the steps of:

2 (b121) obtaining a dependency change from the dependency changes array;

3 (b122) determining whether the dependency change is in a directory in the master  
4 array;

5 (b123) updating the directory in the master array of the dependency change in a  
6 directory of the master array;

7 (b124) adding dependency change to the master array in a new directory if the  
8 dependency change is not in a directory of the master array;

9 (b125) determining if there is another dependency change in the dependency changes  
10 array after either step (b123) or step (b124); and

11 (b126) repeating steps (b121) – (b125) until all dependency changes have been  
12 obtained from the dependency change array.

1 7. A system for optimizing the use of a plurality of processors when compiling a  
2 program in a computer system, the system comprising:

3 means for providing a list of directories and a list of processors;

4 means for assigning a directory to a next available processor in an ordered manner to  
5 allow the next available processor to compile at least one file within the directory; and

6 means for ensuring that the maximum number of directories can be compiled.

1           8.       The system of claim 7 wherein the assigning means further includes means for  
2 obtaining a directory in which all dependencies and prerequisites have been satisfied.

1           9.       The system of claim 7 wherein the assigning means further includes the means  
2 for updating the list of processors and the list of directories based upon the assignment of the  
3 directory.

1           10.      The system of claim 10 wherein the assigning means further includes the means  
2 for providing a directory update mechanism for assigning the directories in the ordered manner.

1           11.      The system of claim 10 wherein the providing an update mechanism means  
2 further comprises:

3               means for providing an array of dependency changes; and

4               means for merging the dependency changes array with a master array of changes.

1           12.      The method of claim 5 wherein the merging means comprises:  
2               means for obtaining a dependency change from the dependency changes array;  
3               means for determining whether the dependency change is in a directory in the master  
4 array;

5               means for updating the directory in the master array of the dependency change in a  
6 directory of the master array;

7               means for adding dependency change to the master array in a new directory if the  
8 dependency change is not in a directory of the master array;

9 means for determining if there is another dependency change in the dependency  
10 changes array; and

11 means for ensuring that all dependency changes have been obtained from the  
12 dependency change array.

1 13. A computer readable medium containing program instructions for optimizing  
2 the use of a plurality of processors when compiling a program in a computer system, the  
3 program instructions for:

- 4 (a) providing a list of directories and a list of processors;  
5 (b) assigning a directory to a next available processor in an ordered manner to  
6 allow the next available processor to compile at least one file within the directory; and  
7 (c) repeating step (b) to ensure that the maximum number of directories can be  
8 compiled.

1 14. The computer readable medium of claim 13 wherein the assigning step (b)  
2 further includes the step of (b1) obtaining a directory in which all dependencies and  
3 prerequisites have been satisfied.

1 15. The computer readable medium of claim 13 wherein the assigning step (b)  
2 further includes the step of (b1), updating the list of processors and the list of directories based  
3 upon the assignment of the directory.

1 16. The computer readable medium of claim 13 wherein the assigning step (b)

2 further includes the step of (b1) providing a directory update mechanism for assigning the  
3 directories in the ordered manner.

1 17. The computer readable medium of claim 16 wherein the providing an update  
2 mechanism step (b1) further comprises the steps of:

3 (b11) providing an array of dependency changes; and

4 (b12) merging the dependency changes array with a master array of changes.

1 18. The computer readable medium of claim 17 wherein the merging step (b12)  
2 comprises the steps of:

3 (b121) obtaining a dependency change from the dependency changes array;

4 (b122) determining whether the dependency change is in a directory in the master  
5 array;

6 (b123) updating the directory in the master array of the dependency change in a  
7 directory of the master array;

8 (b124) adding dependency change to the master array in a new directory if the  
9 dependency change is not in a directory of the master array;

10 (b125) determining if there is another dependency change in the dependency changes  
11 array after either step (b123) or step (b124); and

12 (b126) repeating steps (b121) – (b125) until all dependency changes have been  
13 obtained from the dependency change array.